|  |  |  |  |
| --- | --- | --- | --- |
| **QUEST INTERNATIONAL UNIVERSITY**  **PERAK**      **FACULTY OF SCIENCE AND TECHNOLOGY (FST)**    ACADEMIC YEAR 2019    NOVEMBER 2019    [Bachelor in Computer Science (Hons)](http://www.utar.edu.my/admission_app/201101/B/MER_BUS_FT.htm)    **BCS3105 PROJECT 2**  **PROJECT INTERIM REPORT**    **PREPARED BY:** | | | |
| **Candidate Name** | **LEE ZHENG HUNG** | | |
| **Project Title** | **Intelligent Traffic Light & Surveillance System (recognition Module & API)** | | |
| **Supervisor’s Name** | **DR. NOORHIDAYAH BINTI ZAKARIA** | | |
| **Moderator’s Name** | **MR ALVIN CHEANG KAH WAI** | | |
| **Submission Date** | **28 FEB 2020** | **Submission Time** | **5.00PM** |

**QUEST INTERNATIONAL UNIVERSITY PERAK**

**INTELLIGENT MANAGEMENT AND SURVEILLANCE SYSTEM**

**(recognition Module & API)**

**by**

**Lee Zheng Hung**

The undersigned certify that they have read, and recommend to the Postgraduate Studies Programme for acceptance this thesis for the fulfilment of the requirements for the degree stated.

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Main Supervisor: DR. Noor Hidayah

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Co-Supervisor: Mr. Alvin Cheang Kah Wai

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Head of Programme: Dr. Sharanjit Kaur

Date: 28/2/2020

INTELLIGENT TRAFFIC MANAGEMENT AND

SURVEILLANCE SYSTEM

(**recognition Module & API**)

By

Lee Zheng Hung

A Thesis

Submitted to the School of Computing

as a Requirement for the Degree of

BACHELOR OF COMPUTER SCIENCE (HONs)

QUEST INTERNATIONAL UNIVERSITY PERAK

IPOH,

PERAK

NOVEMBER 2019

**DECLARATION OF THESIS**

|  |  |
| --- | --- |
| Title of thesis | INTELLIGENT TRAFFIC MANAGEMENT AND SURVEILLANCE |
|  |
|  | SYSTEM  (**recognition Module & API**) |
|  |  |

I Lee Zheng Hung hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at QIUP or other institutions.

Witnessed by:

|  |  |  |
| --- | --- | --- |
| **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | |
| Signature of Author | Signature of Supervisor | |
|  |
| Lee Zheng Hung |  |  |
| Permanent address: | Name of Supervisor: | |
|  |
| No 16, |  |  |
| Hala Sepakat 18, |  |  |
| Taman Harmoni, | Dr. Noor Hidayah | |
| 31350, Ipoh, |  |  |
| Perak |  |  |
| Date: - | Date: - | |

**ACKNOWLEDGEMENTS**

First and foremost, I would like to give a heart full of appreciation to my project supervisor, Dr. Noor Hidayah Binti Zakaria for her encouragement and continuous support throughout this project as well as her guidance and feedback on reviewed reports.

Secondly, I would like to give another heart-warming thank you to my moderator Mr. Alvin Cheang Kah Wai for his views and ideas on the downfalls of my project as well as the need to view my project on different perspectives to get a better understanding.

I would like to thank Dr. Sharanjit Kaur, Mr. Yen Pei Tay and Dr. Lee Lam Hong for always providing a positive attitude and a never-ending support.

I would also like to thank to my project partner Mr. Phong Kah Weing for all the support, knowledges and words of wisdom leading up to this project and throughout this project as well.

I would like to thank some of my seniors such as Mr. Seow Si Xiang and Mr. Lian Zheng Hao for sharing their knowledge to me and navigate me through certain complication.

Finally, I would like to thank my classmates and course mates for always being there for me and providing a positive vibe.

A million thank you is not enough for me to express the appreciation for all the help I have received.

**Abstract**

Have you ever felt frustrated with the busy traffic at peak hours? This project aims to ease traffic congestion using low-cost sensors with on-board machine learning techniques. The ultimate goal for this project is to reduce traffic jam at peak hour by tracking known congestion hotspot that will lead to long lines of car. The main focus of this project is to manage the traffic around the junction area. By having the assistance of machine learning technique, the system will be able to recognize the number of vehicles on each side of the road by using data to manage the traffic flow more efficiently by controlling the timer of traffic light. The system is also allowing the connected emergency vehicles such as ambulance or police car get through in the middle of a traffic congestion when they are sending patients to treatment facilities, such as hospitals or respond to incidents and chases. With this system it can reach to their destination quickly and do their duties. The sub function is including with vehicle surveillance to search and locate lost vehicles, road traffic offenders and monitoring of critical traffic events.

**Keywords**: Intelligent Traffic Light, Vehicle Surveillance

Contents

[1.1 System Overview 9](#_Toc33972182)

[1.2 Background Study 10](#_Toc33972183)

[1.3 Problem Statement 10](#_Toc33972184)

[1.4 Significance of Study 10](#_Toc33972185)

[1.5 Objective 11](#_Toc33972186)

[1.6 Limitation (solvable by using specialized cameras) 11](#_Toc33972187)

[1.7 System Scope (“Car recognition & information identification” Module) 11](#_Toc33972188)

[1.8 Proposed Solution 11](#_Toc33972189)

[1.9 Scope of Solution 11](#_Toc33972190)

[1.10 Impact of Work to Society 11](#_Toc33972191)

[1.11 Journey Map 12](#_Toc33972192)

[1.12 Milestone 12](#_Toc33972193)

[1.13 Work Breakdown Structure 15](#_Toc33972194)

[2.1 Preliminary investigation 16](#_Toc33972195)

[2.2 Secondary Investigation 18](#_Toc33972196)

[2.3 Third Investigation 19](#_Toc33972197)

[3.1 Proposed Method 20](#_Toc33972198)

[3.1.1 Scrum Model Phases 21](#_Toc33972199)

[3.2 Project Iteration 22](#_Toc33972200)

[3.2.1 Iteration 1: (sprint 1) 22](#_Toc33972201)

[3.2.2 Iteration 2: (sprint 2) 23](#_Toc33972202)

[3.2.3 Iteration 3: (sprint 3) 24](#_Toc33972203)

[3.2.4 Iteration 4: (sprint 4) 25](#_Toc33972204)

[3.3 Tools for development 26](#_Toc33972205)

[3.4 Requirement 26](#_Toc33972206)

[3.5 Modelling Technique 27](#_Toc33972207)

[3.5.1 Use Case diagram 27](#_Toc33972208)

[3.5.2 Flow Diagram 28](#_Toc33972209)

[3.5.3 Training Flowchart 29](#_Toc33972210)

[3.5.4 Activity Diagram 30](#_Toc33972211)

[3.6 Non-functional requirement 31](#_Toc33972212)

[3.7 Prototype 32](#_Toc33972213)

[Implementation and Testing 35](#_Toc33972214)

[4.1 Implementation Phase 35](#_Toc33972215)

[4.1.1 Data Preparation 36](#_Toc33972216)

[4.1.2 API Server Implementation 37](#_Toc33972217)

[4.1.3 Car plate detection – Structure 39](#_Toc33972218)

[4.1.4 Emergency vehicle event trigger 40](#_Toc33972219)

[4.2 Testing Phases 41](#_Toc33972220)

[4.2.1 Test Planning 42](#_Toc33972221)

[4.2.2 Test Result 46](#_Toc33972222)

[4.3 Test Report 48](#_Toc33972223)

[5.1 Overview Learning 50](#_Toc33972224)

[5.2 Contribution 50](#_Toc33972225)

[5.3 Challenges 50](#_Toc33972226)

[5.4 Summary 51](#_Toc33972227)

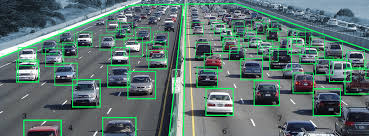
[5.5 Future work 51](#_Toc33972228)

Chapter 1

# 1.1 System Overview

The Title of our project is named “Intelligent Traffic Light & Surveillance System” which allow real time car recognition and car identify the type of car and car plate. The system will store history of car for in a decentralized blockchain database to avoid tempering and exploitation. The main goal of the system is to reduce traffic congestion and allow emergencies vehicle to quickly move through traffic by manipulating the traffic light based on the current traffic. The system will able to determine traffic status and smartly adapt based on the data from live camera, data will also be stored in decentralized blockchain database for integrating other feature. Lastly, Intelligent traffic Control & Surveillance System can track down lost / stolen car, therefore it can be used in assisting police search to increase success rate in tracking down criminal.

For my particular scope which is “Car Detection & information identification” is aim to train a system in recognizing common car in Malaysia. The system needed to be able to clearly define the boundary box of the vehicles in the frame. The system of this module will also include features such as car number plate recognition and store history record of the car for investigation usage. The system will have a backend system and front end API to allow developer to effectively communicate and retrieve information from the API server without having to deploy the module to every single traffic light, increasing the efficiency and making it easier to manage.



# 1.2 Background Study

Based on current Traffic System, how the traffic light turn green is strictly controlled by predetermined timer determined by the developer or needed to be manually configured by the administrator one by one. There are also sensor based traffic light with detect if there is car in the lane, this is usually used in more rural area or road that has less traffic to avoid unnecessary stop when no traffic is present.

Current System also do not use any recognition device, therefore there is no way of determining the exact percentage of traffic. Traffic officer need to be assigned to manage conjected road, this increase the cost and time needed to manage traffic.

# 1.3 Problem Statement

* Not flexible traffic control
* Depended on initial set up
* Not efficient
* Sensor based traffic light not reliable in heavy traffic condition

Weather how the traffic situation is, the current traffic light will only function in a way depend on how the system is set up. Although it is working as it intended which is controlling traffic but it is not 100 percent efficient in managing the traffic flow. Sensor based traffic light is also not reliable in heavier traffic road since it is not reliable when dealing with huge traffic of vehicles. Whether there is huge line of traffic or not, the traffic light will still perform the same way, causing traffic jam on busy hour.

# 1.4 Significance of Study

The benefits of implementing an intelligent traffic control system is to maximize the efficiency of traffic control and reducing traffic jam. It also provide a great source of data for car tracking that can be use for criminal tracking. This module include Machine learning that is used to identify car on the road and perform analysis and providing vital information for both of this vital features and other module.

# 1.5 Objective

**“Intelligent Traffic Management & Surveillance System”**

* Detect Car
* Recognize Car plate
* Recognize Car Type
* Identify road condition

*“Car recognition & information identification”* Module

* Accurately identify vehicle information
* Accurately track and store vehicle history
* Identify car plate number
* API for information retrieval

# 1.6 Limitation (solvable by using specialized cameras)

* Prone to weather disturbance
* High resolution camera equipment
* Accuracy dependent on Training model

# 1.7 System Scope (“Car recognition & information identification” Module)

* Learn how to recognize car and car plate based on dataset and retrieve data from video feed.
* Able to get information by calling API.

# 1.8 Proposed Solution

Creating a Conv and YOLOv3 as the machine learning platform. Additional algorithm will be implemented for

# 1.9 Scope of Solution

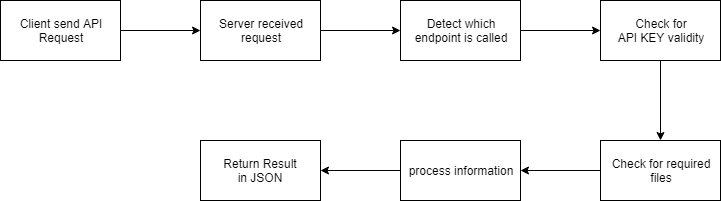
System can be deployed across location for traffic light control. System can be run on different OS such as raspberry pi, windows, IOS, Linux, or any OS that support python.

# 1.10 Impact of Work to Society

This System contribute to:

1. Flexibility for user to manipulate data
2. Can be easily integrated with any system that require the functionality.

# 1.11 Journey Map



# 1.12 Milestone

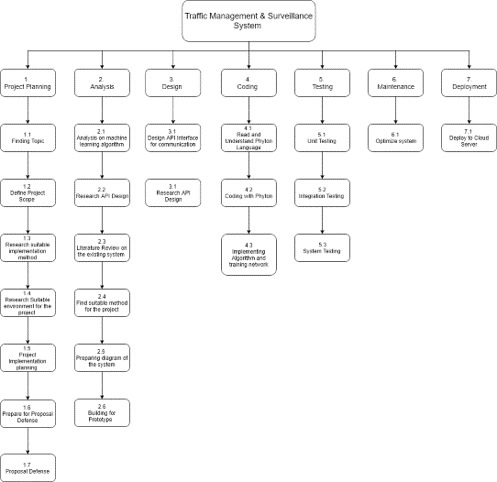
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Name | Duration | Start Date | End Date |
| ***1*** | ***Iteration 1*** | ***48 days*** | ***13/4/2019*** | ***31/5/2019*** |
| **1.1** | **Planning** | **10 days** | **13/4/2019** | **22/4/2019** |
| 1.1.1 | Select a Project Title | 3 days | 13/4/2019 | 15/4/2019 |
| 1.1.2 | Define scope of project | 1 day | 16/4/2019 | 16/4/2019 |
| 1.1.3 | Identify tool and resources | 1 day | 17/4/2019 | 17/4/2019 |
| 1.1.4 | Preparing tool and resources | 3 days | 18/4/2019 | 20/4/2019 |
| 1.1.5 | Prepare project proposal | 2 days | 21/4/2019 | 22/4/2019 |
| 1.1.6 | Proposal defense | 1 day | 23/4/2019 | 23/4/2019 |
| **1.2** | **Analysis** | **13 days** | **24/4/2019** | **6/5/2019** |
| 1.2.1 | Research on car detection | 4 days | 24/4/2019 | 27/4/2019 |
| 1.2.2 | Research on Machine Learning | 5 days | 28/4/2019 | 2/5/2019 |
| 1.2.3 | Literature review on detecton algorithm | 2 days | 3/5/2019 | 3/5/2019 |
| 1.2.4 | Literature review on restful API | 1 day | 4/5/2019 | 4/5/2019 |
| 1.2.5 | Literature review on yolov3 algorithm | 1 day | 5/5/2019 | 5/5/2019 |
| 1.2.6 | Mid-Term Checkpoint | 1 day | 6/5/2019 | 6/5/2019 |
| **1.3** | **Design** | **3 days** | **7/5/2019** | **9/5/2019** |
| 1.3.1 | Design Use Case Diagram | 1 day | 7/5/2019 | 7/5/2019 |
| 1.3.2 | Design Flowchart Diagram | 1 day | 8/5/2019 | 8/5/2019 |
| 1.3.3 | Design Activity Diagram | 1 day | 9/5/2019 | 9/5/2019 |
| **1.4** | **Coding** | **14 days** | **10/5/2019** | **23/5/2019** |
| 1.4.1 | Read and understanding coded algorithm | 1 days | 10/5/2019 | 10/5/2019 |
| 1.4.3 | Building Prototype (/detectCar endpoint) | 9 days | 11/5/2019 | 19/5/2019 |
| 1.4.4 | Building demo interface (client side) | 2 days | 20/5/2019 | 21/5/2019 |
| 1.4.5 | Debuging and handling user error | 1 day | 22/5/2019 | 22/5/2019 |
| 1.4.5 | Final Presentation | 1 day | 23/5/2019 | 23/5/2019 |
| **1.5** | **Testing** | **8 days** | **24/5/2019** | **31/5/2019** |
| 1.5.1 | Functional Testing | 2 days | 24/5/2019 | 25/5/2019 |
| 1.5. | Unit Testing | 2 days | 26/5/2019 | 27/5/2019 |
| 1.5.3 | Integration Testing | 2 days | 28/5/2019 | 29/5/2019 |
| 1.5.4 | System Testing | 2 days | 30/5/2019 | 31/5/2019 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Name | Duration | Start Date | End Date |
| ***1*** | ***Iteration 2*** | ***35 days*** | ***3/7/2019*** | ***3/8/2019*** |
| **1.1** | **Planning** | **2 days** | **3/7/2019** | **4/7/2019** |
| 1.1.1 | Define scope for /detectCarPlate module | 1 days | 3/7/2019 | 3/7/2019 |
| 1.1.2 | Define requirement for selected module | 1 day | 4/7/2019 | 4/7/2019 |
| **1.2** | **Analysis** | **9 days** | **5/7/2019** | **13/7/2019** |
| 1.2.1 | Research on car plate detection | 3 days | 5/7/2019 | 7/7/2019 |
| 1.2.2 | Research on character Segmentation | 3 days | 8/7/2019 | 10/7/2019 |
| 1.2.3 | Research on character recognition | 3 days | 11/7/2019 | 13/7/2019 |
| **1.3** | **Design** | **3 days** | **14/7/2019** | **16/7/2019** |
| 1.3.1 | Design Use Case Diagram | 1 day | 14/7/2019 | 14/7/2019 |
| 1.3.2 | Design Flowchart Diagram | 1 day | 15/7/2019 | 15/7/2019 |
| 1.3.3 | Design Activity Diagram | 1 day | 16/7/2019 | 16/7/2019 |
| **1.4** | **Coding** | **13 days** | **17/7/2019** | **29/7/2019** |
| 1.4.1 | Read and understanding algorithm | 1 days | 17/7/2019 | 17/7/2019 |
| 1.4.3 | Building /detectCarPlate endpoint | 10 days | 18/7/2019 | 27/7/2019 |
| 1.4.4 | Building API connection | 1 days | 28/7/2019 | 28/7/2019 |
| 1.4.5 | Debuging and handling user error | 1 day | 29/7/2019 | 29/7/2019 |
| **1.5** | **Testing** | **8 days** | **30/7/2019** | **6/8/2019** |
| 1.5.1 | Functional Testing | 2 days | 30/7/2019 | 31/7/2019 |
| 1.5. | Unit Testing | 2 days | 1/8/2019 | 2/8/2019 |
| 1.5.3 | Integration Testing | 2 days | 3/8/2019 | 4/8/2019 |
| 1.5.4 | System Testing | 2 days | 5/8/2019 | 6/8/2019 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Name | Duration | Start Date | End Date |
| ***1*** | ***Iteration 3*** | ***35 days*** | ***6/8/2019*** |  |
| **1.1** | **Planning** | **2 days** |  |  |
| 1.1.1 | Define scope for /detectCarType module | 1 days | 6/8/2019 | 6/8/2019 |
| 1.1.2 | Define requirement for selected module | 1 day | 7/8/2019 | 7/8/2019 |
| **1.2** | **Analysis** | **9 days** |  |  |
| 1.2.1 | Research on classification | 3 days | 8/8/2019 | 10/8/2019 |
| 1.2.2 | Research on feature recognition | 3 days | 11/8/2019 | 13/8/2019 |
| 1.2.3 | Research on car type recognition | 3 days | 14/8/2019 | 16/8/2019 |
| **1.3** | **Design** | **3 days** |  |  |
| 1.3.1 | Design Use Case Diagram | 1 day | 17/8/2019 | 17/8/2019 |
| 1.3.2 | Design Flowchart Diagram | 1 day | 18/8/2019 | 18/8/2019 |
| 1.3.3 | Design Activity Diagram | 1 day | 19/8/2019 | 19/8/2019 |
| **1.4** | **Coding** | **13 days** |  |  |
| 1.4.1 | Read and understanding algorithm | 1 days | 20/8/2019 | 20/8/2019 |
| 1.4.3 | Building /detectCarType endpoint | 10 days | 21/8/2019 | 30/8/2019 |
| 1.4.4 | Building API connection | 1 days | 31/8/2019 | 31/8/2019 |
| 1.4.5 | Debuging and handling user error | 1 day | 1/9/2019 | 1/9/2019 |
| **1.5** | **Testing** | **8 days** |  |  |
| 1.5.1 | Functional Testing | 2 days | 2/9/2019 | 3/9/2019 |
| 1.5. | Unit Testing | 2 days | 4/9/2019 | 5/9/2019 |
| 1.5.3 | Integration Testing | 2 days | 6/9/2019 | 7/9/2019 |
| 1.5.4 | System Testing | 2 days | 8/9/2019 | 9/9/2019 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Name | Duration | Start Date | End Date |
| ***1*** | ***Iteration 4*** | ***35 days*** |  |  |
| **1.1** | **Planning** | **2 days** |  |  |
| 1.1.1 | Define scope for /detectTraffic module | 1 days | 9/9/2019 | 9/9/2019 |
| 1.1.2 | Define requirement for selected module | 1 day | 10/9/2019 | 10/9/2019 |
| **1.2** | **Analysis** | **9 days** |  |  |
| 1.2.1 | Research on lane detection | 3 days | 11/9/2019 | 13/9/2019 |
| 1.2.2 | Research on lane analysis | 3 days | 14/9/2019 | 16/9/2019 |
| 1.2.3 | Research on crossing line | 3 days | 17/9/2019 | 19/9/2019 |
| **1.3** | **Design** | **3 days** |  |  |
| 1.3.1 | Design Use Case Diagram | 1 day | 18/9/2019 | 18/9/2019 |
| 1.3.2 | Design Flowchart Diagram | 1 day | 19/9/2019 | 19/9/2019 |
| 1.3.3 | Design Activity Diagram | 1 day | 20/9/2019 | 20/9/2019 |
| **1.4** | **Coding** | **13 days** |  |  |
| 1.4.1 | Read and understanding existing algorithm | 1 days | 21/9/2019 | 21/9/2019 |
| 1.4.3 | Building /detectTraffic endpoint | 10 days | 22/9/2019 | 1/10/2019 |
| 1.4.4 | Building API connection | 1 days | 2/10/2019 | 2/10/2019 |
| 1.4.5 | Debuging and handling user error | 1 day | 3/10/2019 | 3/10/2019 |
| **1.5** | **Testing** | **8 days** |  |  |
| 1.5.1 | Functional Testing | 2 days | 4/10/2019 | 5/10/2019 |
| 1.5. | Unit Testing | 2 days | 6/10/2019 | 7/10/2019 |
| 1.5.3 | Integration Testing | 2 days | 8/10/2019 | 9/10/2019 |
| 1.5.4 | System Testing | 2 days | 10/10/2019 | 11/10/2019 |

# 1.13 Work Breakdown Structure



2.0 Chapter 2

# 2.1 Preliminary investigation

Based on studied jurnoul “Object Detection with Deep Learning: A Review” by Zhong-Qiu Zhao and Shou-tao Xu, they describe the object detection’s close relationship with video analysis and image understanding. Based on the paper, they state that traditional object detection method are handcrafted and not reliable to use in real world usage. The performance of object detection is based on constructing complex ensembles which combines multiple low-level image features with high-level context from object detectors and scene classifiers. With the advancement in deep learning, more powerful and algorithm were developed to improve machine learning to solve these complicated problems. The paper show the review of history of deep learning and representative tool such as Convolutional Neural Network (CNN) and typical generic object detection architectures.

To understand the object in an image, we should not only classify different image, but we need to precisely locate and differentiate different type of object contained in each image. To tackle problem such as face detection, pedestrian detection, skeleton detection and more, the fundamental computer vision problem need to have a object detection that are able to provide valuable information for semantic understanding of the image or videos. This include image classification, human behavior analysis, face recognition and face recognition and autonomous driving. By Inheriting from neural network and related learning system, the progress of neural network algorithms based on learning data will lead to great impact on object detection techniques.

The object detection is define as where object are located in given image. The pipeline of traditional object detection model can be divided into three stages.

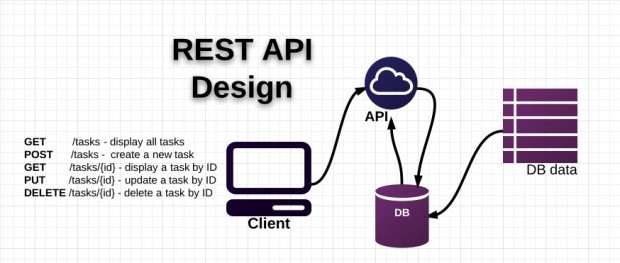
* Informative region selection
  + Scan the whole image with a multi-scale sliding window
* Feature extraction
  + Extract visual features which can provide a semantic and robust representation.
* Classification
  + Distinguish a target object from all the other categories and make the representations more hierarchical, semantic and informative for visual recognition.

The proposal of R-CNN model improved model which optimized classification and bounding box regression task, it takes additional sub-network to generate region proposals and YOLO which accomplish object detection via fixed-grid regression. Each CNN layer is known as a feature map. The Feature map of the input layer is a 3D matrix of pixel intensities for different color channel. The multi-channel image is viewed as specific feature for the model to learn the model.

# 2.2 Secondary Investigation

Based on research paper “APIs and Restful APIs by Mithilesh Tarkar and Ameya Parker, the research paper show detailed explanation about Restful API and Rest APIs. Restful APIs and APIs have become an essential tool when come in backend development. The usage of API is to fetch data and submitting data from client to backend server for processing. With an API, the Developer only has to design the front end of the system whether in machine based, android based or other operating system. The backend of the application will be performing all the calculation and processing and the result will be returned.

API stand for Application Programming interfaces and are helpful for centralized backend system for mobile and web application, it is also widely used for utilization of same function for client running on multiple operating system. When using a API, user send request then information will be fetch from server, or user can send data with JSON request for processing.



# 2.3 Third Investigation

Based on research paper “Evaluation of Haar Cascade classifier for Face Detection”, the author wrote about a method for detecting feature of faces using cascade method. Feature extraction can be divided into 3 approaches: holistic, feature-based and hybrid. Principal component analysis fisher discriminant analysis and support vector machine are examples of holistic approach. Face detection has been improved in terms of speed with the application of haar-features with the contribution of the ViolaJones object detection framework.

How this method work is by having a face database that contains facial image of many individual with different expression. For example sad, sleepy, surprised, wink and with glasses. Then an Integral images can be created, using adaboost training method, we can generate cascading classifiers which is a .xml file that can be used to compare with your dataset.

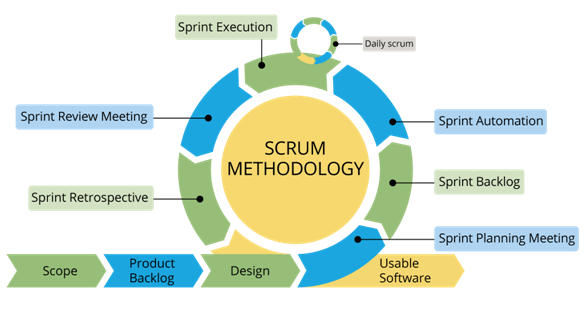
To understand an object in the provided image, the haar cascade dataset must be consistent and accurate. Not only face detection, haar cascade also can be used to detect other object such as car detection. By training a set of pre-selected car image, it can be trained to recognize car inside a frame.

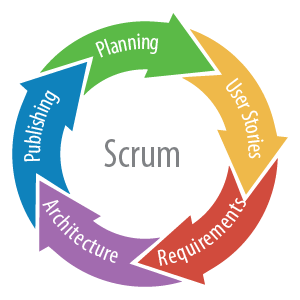
Sample Face Dataset Sample Car dataset

Chapter 3

# 3.1 Proposed Method



****

In this project, the selected methodology is scrum from agile principle since one module can be finish and tested first before starting another phase. This will allow project flow to be modular and make it easier to to add functionality and modify the program based on requirements. Another advantages is to ensure the team do not run into bottleneck and interruption with another part of the program. Using agile model, planning and design will be done when starting a new phase, therefore increasing the reliability of the system. The method that are chosen for agile method is scrum, the scrum model synchronizes the work of team members as they discuss the work of the sprint.

## 3.1.1 Scrum Model Phases

**Step 1. Product Backlog Creation**

Product backlog is a list that consists of features that should be implemented during the development process. It’s ordered by priority and its every item is called a User story, and every user story have a unique ID.

For example:

|  |  |
| --- | --- |
| ID | User Story |
| A-001 | Task 1 |
| A-002 | Task 2 |

**Step 2. Sprint Planning and Sprint Backlog Creation**

Determine the sprint duration, short sprint allows developer to release the working version of a product more frequently. Then, select user stories from the backlog table and decide a method to solve the task given.

**Step 3. Working on the Sprint. Scrum Meetings**

After actual user stories for the current phase are chosen, the system development process begins. There are another important feature for scrum meetings, it allow development process to be accelerated and allow understanding to the task.

**Step 4. Testing and Product Demonstration**

Using fill life-cycle testing process, result of sprint is demonstrated. The Scrum team creates a review and demonstrates the results of their work. On this basis the stakeholders take a decision about further project changes.

**Step 5. Retrospective and Next Sprint Planning**

Discuss the results and determine the way to improve the process on next step. The team should conclude what went well during the working process and what can be done better during the future iteration. When the ways of improvement are defined, the team can concentrate on the next sprint planning.

# 3.2 Project Iteration

## 3.2.1 Iteration 1: (sprint 1)

**Planning**

/detectCar endpoint chosen for sprint 1

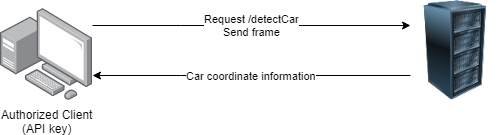
User stories (Backlog Creation)

|  |  |
| --- | --- |
| ID | User Story |
| A001 | Ability to receive two information, API\_KEY and file |
| A002 | System verify API\_KEY validity |
| A003 | System verify valid file type |
| A004 | System process car in frame |
| A005 | System return Information |

**Requirements**

Able to detect car and return information such as coordinate of car in a frame

**Architecture**

****

**Publishing**

Functional testing will be done to ensure all possible error are successfully handled and can be return by server, /detect car endpoint will able to get frame from requester and process the car location inside a frame and return the coordinate information.

## 3.2.2 Iteration 2: (sprint 2)

**Planning**

/detectCarPlate endpoint chosen for sprint 2

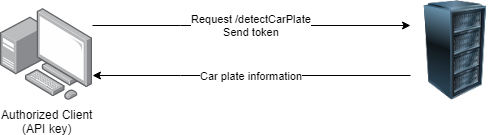
User stories (Backlog Creation)

|  |  |
| --- | --- |
| ID | User Story |
| B001 | Ability to receive two information, API\_KEY and a token |
| B002 | Identify token association with stored car detail |
| B003 | Identify car plate from car |
| B004 | Identify car plate number from car plate |
| B005 | System return car plate information |

**Requirements**

Able to Identify car plate from stored car information and identify the car plate

**Architecture**

****

**Publishing**

Functional testing will be done to ensure all possible error are successfully handled and can be return by server, /detectCarPlate endpoint will able to get token from requester and Identify car plate and return the car plate information.

## 3.2.3 Iteration 3: (sprint 3)

**Planning**

/detectCarType endpoint chosen for sprint 3

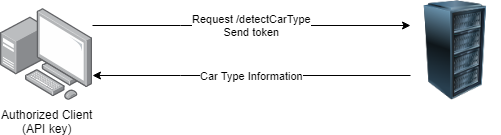
User stories (Backlog Creation)

|  |  |
| --- | --- |
| ID | User Story |
| B001 | Ability to receive two information, API\_KEY and a token |
| B002 | Identify token association with stored car detail |
| B003 | Identify car type from frame |
| B004 | System return car type information |

**Requirements**

Able to identify car type from existing stored car information

**Architecture**



**Publishing**

Functional testing will be done to ensure all possible error are successfully handled and can be return by server, /detectCarType endpoint will able to get token from requester and process the car type from existing car database and return the car type information.

## 3.2.4 Iteration 4: (sprint 4)

**Planning**

/detectTrafficStatus endpoint chosen for sprint 4

User stories (Backlog Creation)

|  |  |
| --- | --- |
| ID | User Story |
| B001 | Ability to receive two information, API\_KEY and a frame |
| B002 | Identify frame and detect the incoming car and outgoing car |
| B003 | Calculate the percentage |
| B004 | System return traffic status information |

**Requirements**

Able to identify traffic status for incoming car and outgoing car from given frame

**Architecture**

****

**Publishing**

Functional testing will be done to ensure all possible error are successfully handled and can be return by server, /detectTrafficStatus endpoint will able to get frame from requester and process the traffic status and return the information to the requester.

# 3.3 Tools for development

* Draw.io
  + Tools for drawing flowchart, Activity diagram, Use Case, and other System diagram
* Gantt project
  + Tools for scheduling timeline and chart out gantt chart for selected project
* Pycharm IDE
  + Python IDE for project development
* OpenCV
  + Image processing library
* Flask
  + API handling library for python
* PyQt5
  + Quick UI prototyping to test API
* Tensorflow
  + Dataset Training and classification
* Postman
  + API Testing tool, request and receiving information
* Python 3.7
  + Environment choose for project development

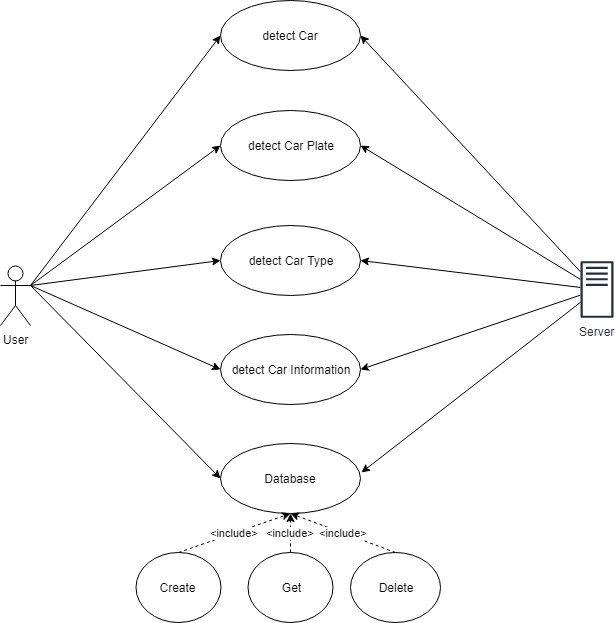
# 3.4 Requirement

Hardware requirement

* Any operating system can support python 3.7 environment
* Capable CPU and GPU for image processing
* 10mbps or higher stable upload and download connection for sending information

# 3.5 Modelling Technique

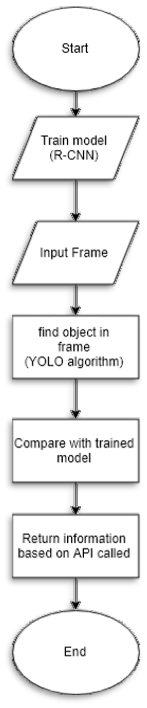
## 3.5.1 Use Case diagram



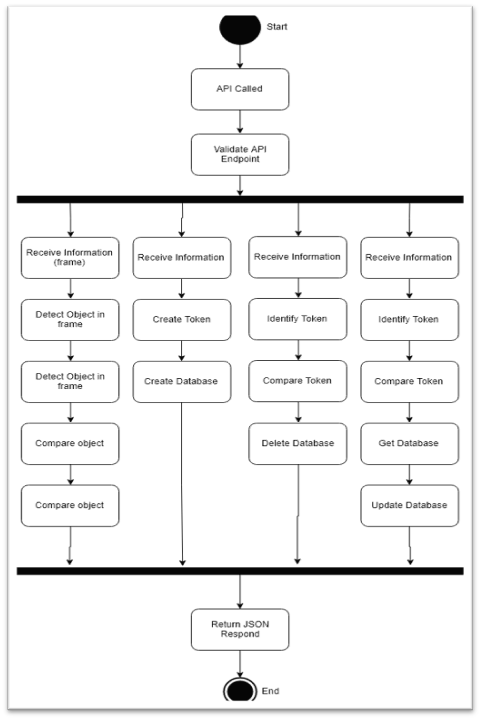
## 3.5.2 Flow Diagram



## 3.5.3 Training Flowchart



## 3.5.4 Activity Diagram



# 3.6 Non-functional requirement

Usability

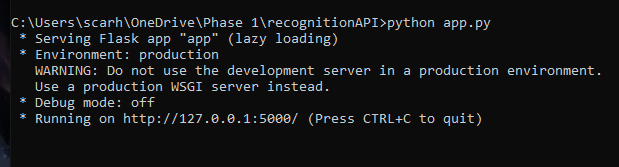
* Returned data can be easily use for any programming language
* Clearly defined endpoint and parameter key for requesting API

Efficiency of System

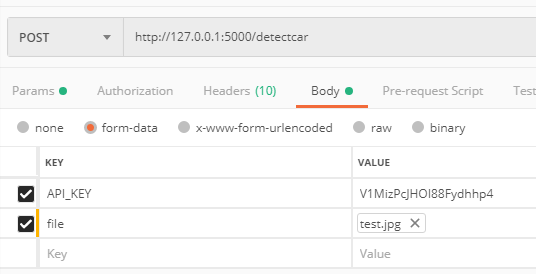
* Auto purge database to reduce storage use
* Can receive file like photo

# 3.7 Prototype

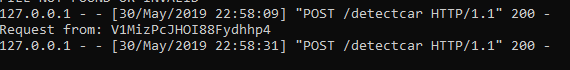
Starting the server



Requesting /detectCar endpoint with postman (client side)



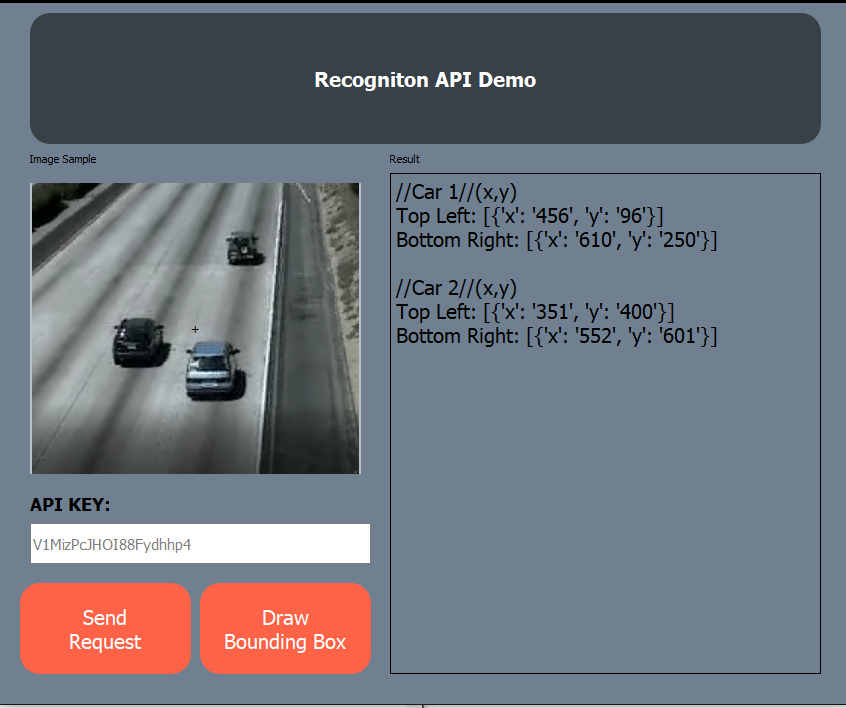
Incoming Request from user (server side)



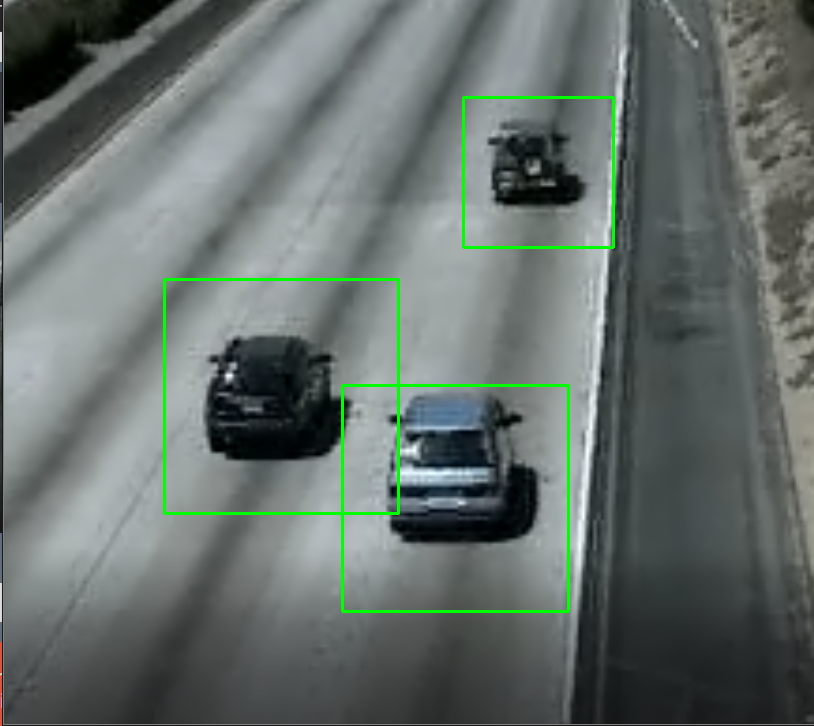
Information Received in JSON format



UI Demonstration for API prototyping

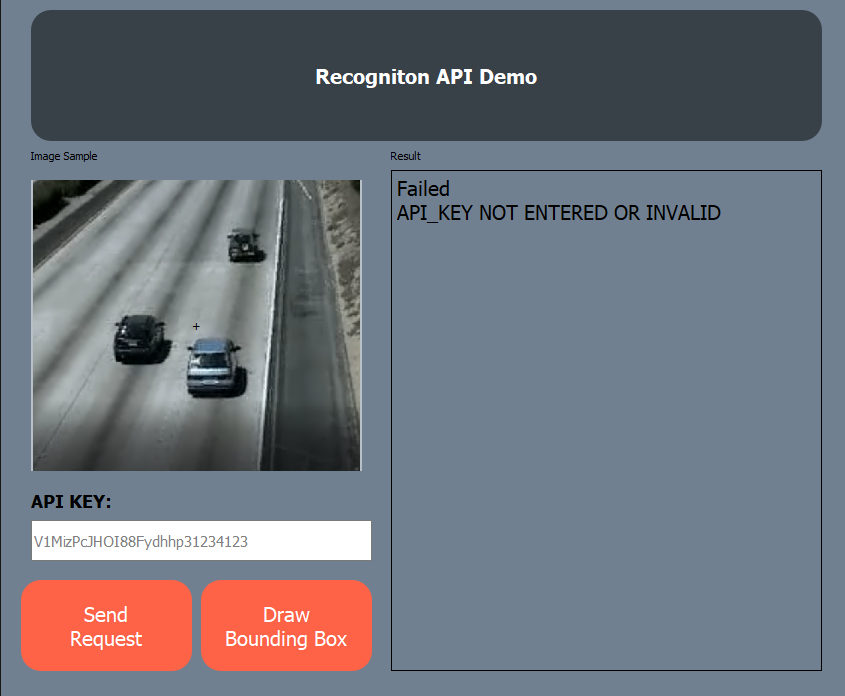


Based on Returned information, client can use the information for further analysis



API Handling user error

Unauthorized API key Incorrect format or no file sent

Chapter 4:

# Implementation and Testing

Testing phase of the software development life cycle (SDLC) is where developers focus on investigate and discovery. In this testing phase, functionalities and feature will be tested extensively to determine whether the software is working as intended and met the requirements. This chapter will break down into two sub section which is implementation phase and testing phase. In Implementation phase, step by step implementation of detection algorithm will be explained with examples. In testing phase, all known use case of the program will be tested intensively to reduce the number of error and minimize chances of software failure when release.

# 4.1 Implementation Phase

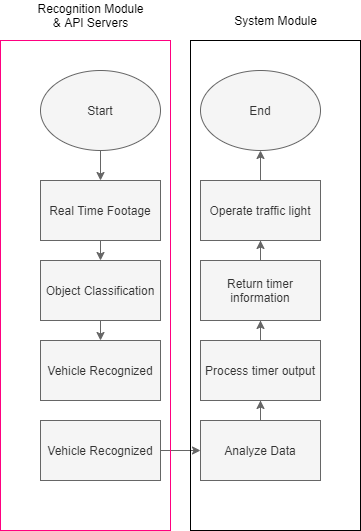


Figure 4.1 General Flow of Intelligent Traffic management and Surveillance System

## 4.1.1 Data Preparation

In recognition module, YOLO Framework is chosen as the model which the detection algorithm runs on. To perform an accurate detection, a good network must be used to get a good result. This start with data preparation, high resolution image is needed to preserve as much detail as possible.

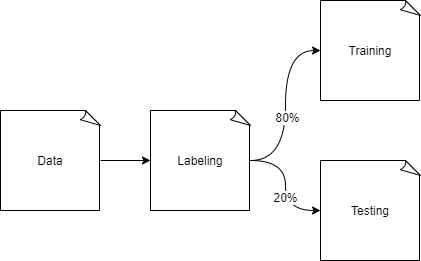


Figure 4.1.1.1 Data Preparation

In data preparation, high resolution images is selected from variety of sources. Including video, google images, vehicle website etc.. . Then, each vehicle in the image is labelled into their own category. The data is then split into two folder for training and testing. It is important to use different data from the training set for testing to determine the accuracy of trained model.

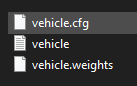


Figure 4.1.1.2 Output File

Once Training are finish, we ended up with a .weight file which represent the network. A vehicle.cfg file is created for all the settings of the network, and a .txt file containing all the class label.

## 4.1.2 API Server Implementation

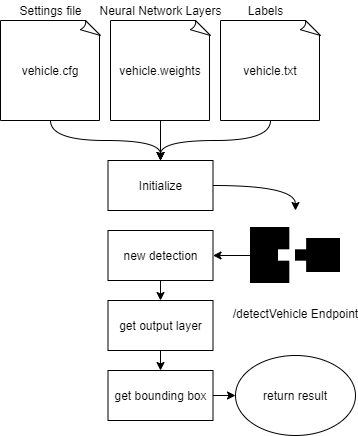


Figure 4.1.2.1 API Implementation diagram

Figure 4.1.2.1 show the visualized algorithm for the API server. Trained network is loaded into the main program and initialized. Then API endpoint is initiated with **Flask**(Python API supporting library). Once the endpoint is called, it will carry 1 image/frame and this new data will be loaded into the network, after getting output layer and comparing, bounding box is acquired and result are returned to the client.



Figure 4.1.2.2 Client Side Data



Figure 4.1.2.3 Server Side Data

Before the image is send to the server, some cleaning is done to exclude all the unwanted data. Figure 4.1.2.2 and 1.1.2.3 shows the view of data from client side and server side respectively.

## 4.1.3 Car plate detection – Structure

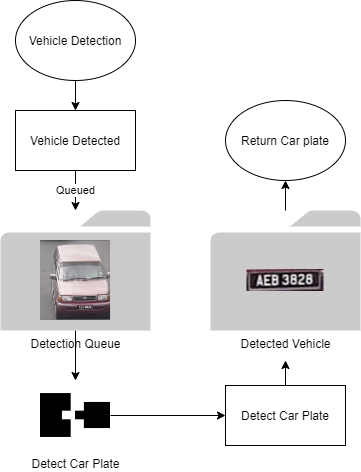
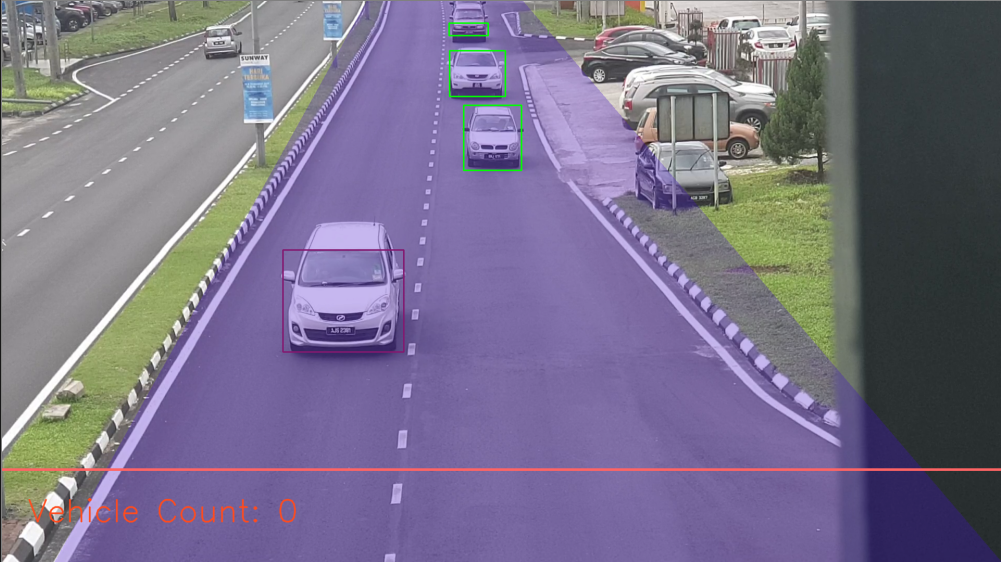


Figure 4.1.3.1

Figure 4.1.3.1 show visualized algorithm for car plate detection algorithm, once vehicle is detected in the frame, it will be saved into a queue folder. Then car plate detection will be performed on these images one by one, the result is will be stored while car plate location in the frame and car plate number will be returned to the client.

## 4.1.4 Emergency vehicle event trigger



Simulated Emergency Vehicle

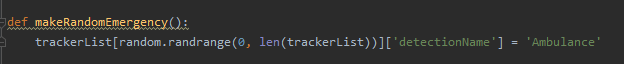


Figure 4.1.4.1

A simple function is created to make one of the vehicle detected become ambulance to simulate emergency vehicle on road for testing purposed.

Once Emergency pass the orange line, event will be triggered to update traffic light, the counter of the traffic light is determined by algorithm on traffic management part.

## 4.2 Testing Phases

Testing phase starts with test plan which will outline the type of testing used, resources used for testing and how the software will be tested. After the test plan has been developed, each test cases are created through the test of each component of the test cases to massively remove errors and potential failures. The purpose of the test cases is to help developers understand more on the limitations and weakness of the system. All test plan is performed after the integration of all part of the system.

Figure 4.2.1

|  |  |
| --- | --- |
| Testing Phases | |
| Test Planning | Document that descript project scope and the event. Test plan that this project use is Testing Level Specific Test Plan. It describes each level of testing such as Unit Test plan, Integration Test Plan, System Test Plan and Acceptance Test Plan. |
| Test Case Development | Once Test Planning is ready, development of the test cases is initiated. Main objective of test case development is to prepare test cases for an individual unit. |
| Test Execution | Test execution is the process of executing the code and comparing the expected and actual results. |
| Test Reporting | Test Report is needed to reflect testing results in a formal way, which gives an opportunity to estimate testing results quickly. |

### 4.2.1 Test Planning

#### 4.2.1.1 Unit Testing

Unit testing is done on every individual unit and component of the software to make sure it performs as designed. Table below shows the components that were tested during unit testing phase. Unit Testing is a kind of white box testing, testing for every possible feature within a program.

API server (Vehicle)

|  |  |  |
| --- | --- | --- |
| Unit Testing | | |
| Receive request | Classification and labelling | Return Request |
| * Able to receive request from client through /detectVehicle endpoint * Receive image and able to store inside server folder * Successfully load image into neural network * Handle incorrect file type * Handle unexpected datatype * Handle invalid key * Handle server crashing, output to console | * Get all the output layer from neural network * Get bounding box for detected object * Classify bounding box for with label * Get location of bounding box in the image | * Package all information including label, bounding box location and confidence in dictionary format * Convert stored dictionary to JSON format * Return the JSON output to client |

API server (Car Plate)

|  |  |  |
| --- | --- | --- |
| Unit Testing | | |
| Receive request | Classification and labelling | Return Request |
| * Able to receive request from client through /detectCarPlate endpoint * Receive image and able to store inside server folder * Successfully load image into neural network * Handle incorrect file type * Handle unexpected datatype * Handle invalid key * Handle server crashing, output to console | * Get all the output layer from neural network * Get bounding box for detected object * Classify bounding box with car plate number with confidence * Get location of bounding box in the image | * Package all information including label, bounding box location and confidence in dictionary format * Convert stored dictionary to JSON format * Return the JSON output to client * Return Error code if not valid input value detected |

Every single unit above is tested to ensure API server can handle all user input without error or crashes.

#### 4.2.1.2 Integration Testing

Integration Testing combines each component and tested as “subsystems”. The goal is to expose fault when in the interaction between units. Integration testing is a kind of black box testing, accept user input, and expect a output without considering how the algorithm work

API Server (Vehicle Detection)

|  |  |  |
| --- | --- | --- |
| Integration Testing | | |
| Receive request | Classification and labelling | Return Request |
| * Able to receive request from client through /detectVehicle | * Get all bounding box and label it with detected car type with location | * Return JSON output with bounding box location, car type and confidence. |

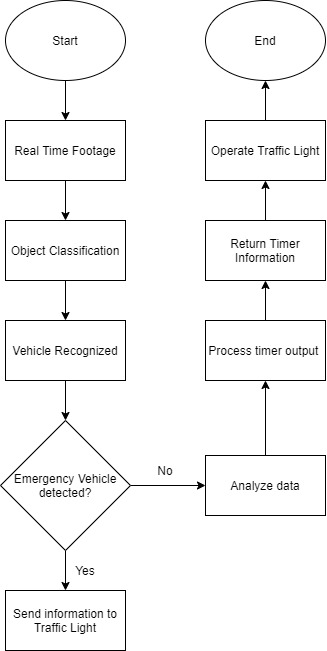
API Server (Car Plate Detection)

|  |  |  |
| --- | --- | --- |
| Integration Testing | | |
| Receive request | Classification and labelling | Return Request |
| * Able to receive request from client through /detectCarPlate | * Get all bounding box and label it with detected car type with location | * Return JSON output with bounding box location, car type and confidence. |

All possible input is send to the server, and desire output is tested and verified.

#### 4.2.1.3 System Testing

System testing combines all the unit to become a whole system, then test is done on the complete system. The purpose of system testing is to evaluate the system’s compliance with the specified requirements or not. In this phase, system module will integrate with Traffic Management and Surveillance module and test on the whole system and see whether met the requirement and follow the general flow



### 4.2.2 Test Result

#### 4.2.2.1 Request receive from client



Figure 4.2.2.1.1 Requet detected

Figure 4.2.2.1.1 show that request is received from client with IP 127.0.0.1 at /detectCar endpoint.

#### 4.2.2.2 Handle request with invalid parameters



Figure 4.2.2.2.1



Figure 4.2.2.2.2

Figure 4.2.2.1 show incorrect file type detected, the server only accepts png, jpg, and byte format. Figure 4.2.2.2 show incorrect key. If error is intercepted, detection will not run and error code returned.

#### 4.2.2.3 Save received image into folder



Figure 4.2.2.3.1

Figure 4.2.2.3.1 show image successfully saved from client request.

#### 4.2.2.4 Save received detected vehicle (car plate module)



Figure 4.2.2.4.1

Figure 4.2.2.4.1 show image successfully saved from client request.

#### 4.2.2.5 Car plate detection and labelling



Figure 4.2.2.5.1

Figure 4.2.2.5.1 show car plate and bounding box detected.

## 4.3 Test Report

|  |  |  |  |
| --- | --- | --- | --- |
| Test Report | | | |
| Test ID | Test Case | Expected Result | Test Result |
| STD\_TC\_01 | Receive request from client (/detectVehicle) | Show ip from user with called endpoint | ✔ |
| STD\_TC\_02 | Save image requested by user (/detectVehicle) | Save .png file to .jpg | ✔ |
| Save .jpg file to .jpg | ✔ |
| Save byte into .jpg | ✔ |
| Return error for non .png/.jpg/byte format request | ✔ |
| STD\_TC\_03 | Bounding Box Detection | Detect Van Bounding Box | ✔ |
| Detect Motorcycle Bounding Box | ✔ |
| Detect Car Bounding Box | ✔ |
| Detect Truck Bounding Box | ✔ |
| Detect Ambulance Bounding Box | ✔ |
| STD\_TC\_04 | Vehicle Label Detection | Label Van | ✔ |
| Label Motorcycle | ✔ |
| Label Car | ✔ |
| Label Truck | ✔ |
| Label Ambulance | ✔ |
| STD\_TC\_05 | Package Information into JSON (vehicle detection) | Store Vehicle Type | ✔ |
| Store Bounding Box Location (top left, bottom right) | ✔ |
| Store confidence score | ✔ |
| STD\_TC\_06 | Return json result (vehicle detection) | Client receive json result | ✔ |
| STD\_TC\_07 | Receive request from client (/detectCarPlate) | Show ip from user with called endpoint | ✔ |
| STD\_TC\_08 | Save image requested by user (/detectCarPlate) | Save .png file to .jpg | ✔ |
| Save .jpg file to .jpg | ✔ |
| Save byte into .jpg | ✔ |
| Return error for non .png/.jpg/byte format request | ✔ |
| STD\_TC\_09 | Car plate bounding box detection | Detect location of car plate bounding box | ✔ |
| STD\_TC\_10 | Car plate label detection | Label bounding box with detected car plate | ✔ |
| STD\_TC\_11 | Package Information into JSON (car plate) | Store car plate and confidence | ✔ |
| STD\_TC\_12 | Return json result (car plate) | Client receive json result | ✔ |

Chapter 5:

Discussion and analysis

# 5.1 Overview Learning

The learning experience that I have obtained from this project is how the fundamentals of machine learning works. By using machine learning technique, vehicle can be accurately detected which open up a lot of possibility in modern system development and design. Comparing to traditional method of object detection which uses feature matching, machine learning model is far more accurate when preforming detection in various lighting condition and weather conditions. From the understanding of current traffic light system, the challenges and constraints can be observed. Therefore, with the technology available now, we are able to come up with innovative idea and solution in traffic management system by utilizing machine learning approach.

# 5.2 Contribution

The system module of Intelligent Traffic Management and Surveillance System will contribute to solve the issues we face every day such as traffic jam during peak hour. Since the system are using machine learning approach, we are able to extend the functionality which is emergency vehicle detection. Therefore, when emergency vehicle are on their duty and are facing challenging traffic condition, the developed system can aid the on duty emergency vehicle to cross the road safely. The developed system will also decrease traffic jam issue by analysing traffic pattern and predict time interval needed for the traffic light to update in order to reduce traffic jam. While doing this, it also records the type of vehicles for example, van, truck, car, motorcycle and ambulance. These data will be stored inside a database and can be use to improve future prediction. The advantages of this approach is the model will keep learning by itself as time goes, unsupervised learning.

# 5.3 Challenges

The challenges faced while developing Intelligent Traffic Management and Surveillance System:

* Accuracy of machine learning model

Since “high quality” data is needed for training a model, it is hard to determine how the condition of data is needed. Therefore, there is a lot of trial and error in determining the best approach on training our model.

* Python library integration  
  A lot of new library that we have not learn before need to be integrated together. Hence, it is unavoidable to run into a lot of bugs and unexpected errors that we need to solve.

# 5.4 Summary

All in all, it can be concluded that the Intelligent Traffic Management and Surveillance System able to change the way of the traffic light function combine with Artificial Intelligent. It can enable smart and cost-effective handling of traffic flow to decrease the traffic jam problem by sense, analyse, store and predict traffic patterns, as well as optimize traffic light systems through an automated control centre.

# 5.5 Future work

As connected vehicles will be taking over the road in the future, the self- driving vehicles software should be able to communicate with an intelligent traffic light and surveillance system to enable a smart and safe traffic management without using with the camera. Other that than, user can set their destination and routes on the connected vehicle and the system will be able to know how many cars will be pass routes. If certain amount of user selects the route, it will advise the connected vehicle to use other routes or make the time interval of the traffic light longer to avoid traffic jam.